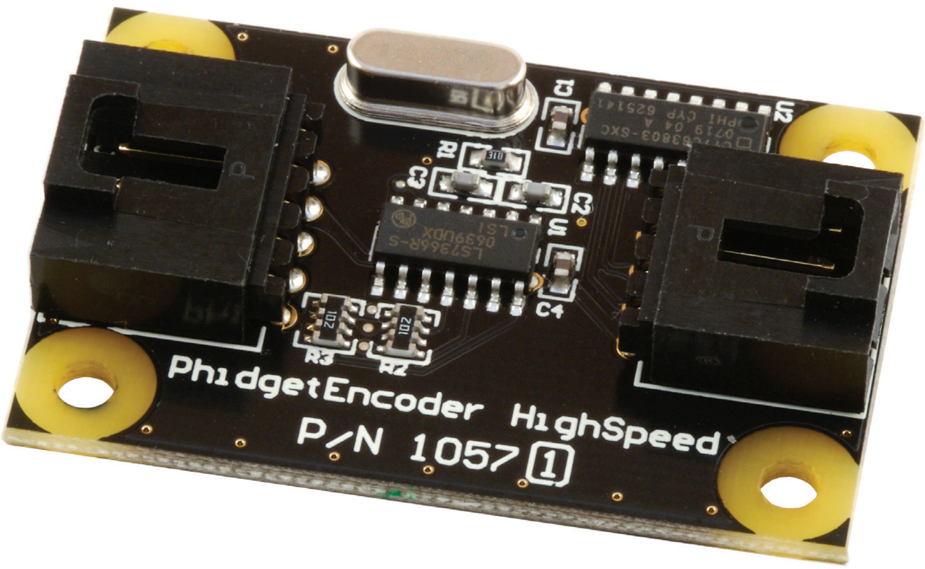
Product Manual

1057 - PhidgetEncoder HighSpeed



Phidgets 1057 - Product Manual

For Board Revision 1

Contents

12 Product History

12 Support

Product Features

- Connects to any typical two-bit optical encoder.
- Returns up to 500,000 counts per second.
- Detect changes in incremental position
- Easily track changes with respect to time
- Connects to almost any optical encoder to measure shaft speed
- Can be used as an accurate human input device.

Programming Environment

Operating Systems: Windows 2000/XP/Vista/7, Windows CE, Linux, and Mac OS X

Programming Languages (APIs): VB6, VB.NET, C#.NET, C++, Flash 9, Flex, Java, LabVIEW, Python, Max/MSP, and Cocoa.

Examples: Many example applications for all the operating systems and development environments above are available for download at www.phidgets.com >> Programming.

Connection

The board connects directly to a computer's USB port.

Getting Started

Checking the Contents

You should have received:

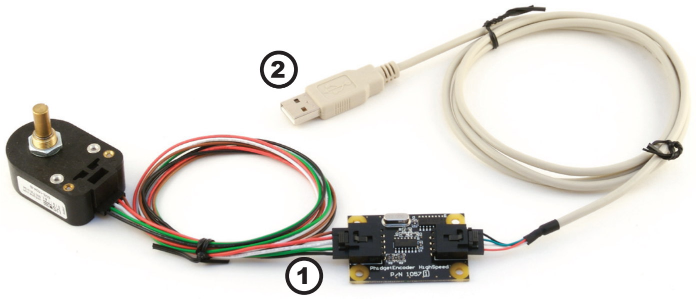
- A PhidgetEncoder HighSpeed board
- A custom USB cable
- A 5-wire encoder cable

In order to test your new Phidget you will also need:

- An Incremental Encoder with Quadrature output (and known pinout)

Connecting all the pieces

1. Connect your Encoder to the PhidgetEncoder board using the 5-wire cable.
2. Connect the PhidgetEncoder board to your computer using the custom USB cable.



Testing Using Windows 2000/XP/Vista/7

Downloading the Phidgets drivers

Make sure that you have the current version of the Phidget library installed on your PC. If you don't, do the following:

Go to www.phidgets.com >> Drivers

Download and run Phidget21 Installer (32-bit, or 64-bit, depending on your PC)

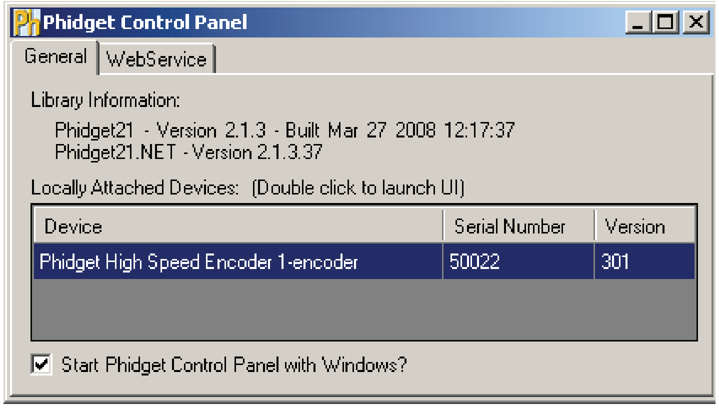
You should see the Ph icon on the right hand corner of the Task Bar.

Running Phidgets Sample Program

Double clicking on the Ph icon loads the Phidget Control Panel; we will use this program to make sure that your new Phidget works properly.

The source code for the Encoder-Full sample program can be found under C# by clicking on Phidget.com > Programming.

Double Click on the Ph icon to activate the Phidget Control Panel and make sure that the **Phidget High Speed Encoder** is properly attached to your PC.



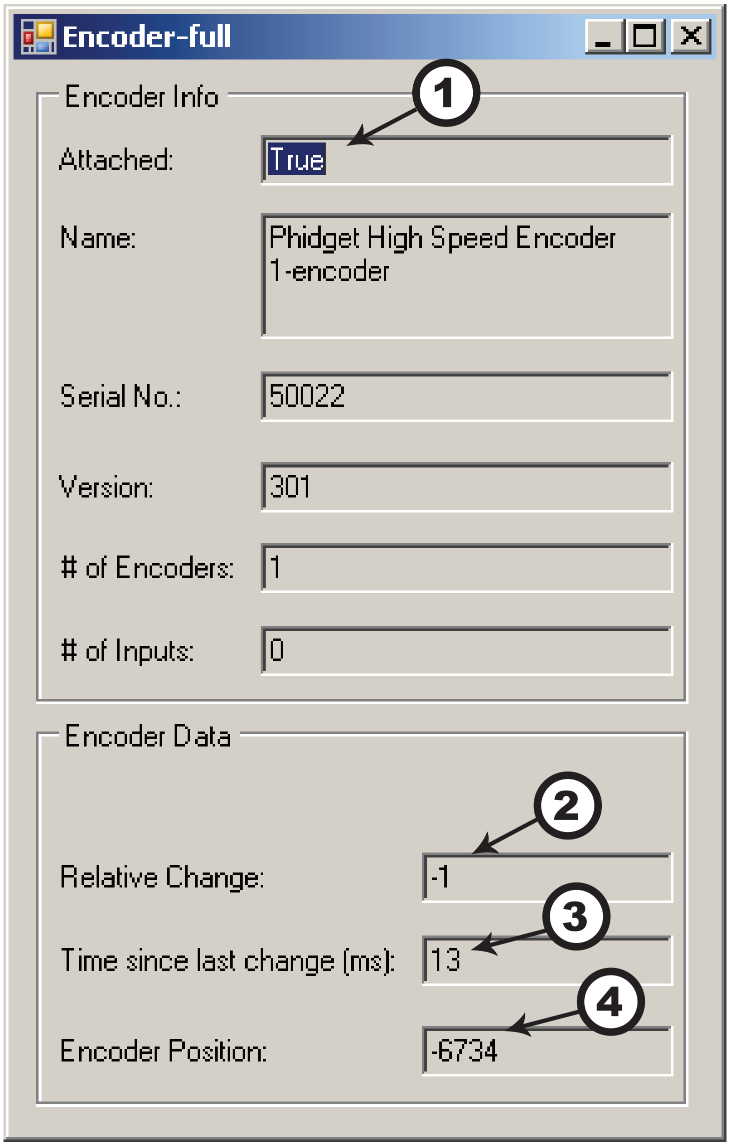
Library Information:

Phidget21 - Version 2.1.3 - Built Mar 27 2008 12:17:37
Phidget21.NET - Version 2.1.3.37

Locally Attached Devices: (Double click to launch UI)

Device	Serial Number	Version
Phidget High Speed Encoder 1-encoder	50022	301

☑ Start Phidget Control Panel with Windows?



Encoder Info

Attached:	True
Name:	Phidget High Speed Encoder 1-encoder
Serial No.:	50022
Version:	301
# of Encoders:	1
# of Inputs:	0

Encoder Data

Relative Change:	-1
Time since last change (ms):	13
Encoder Position:	-6734

1. Double Click on **Phidget High Speed Encoder** in the Phidget Control Panel to bring up Encoder-full and check that the box labelled Attached contains the word True.
2. As you turn the knob the relative change will be positive when turning counterclockwise and negative when turning clockwise.
3. Displays the time in milliseconds since the last time you turned the knob.
4. The number will decrease when turning the knob counterclockwise and increase when turning clockwise.

Testing Using Mac OS X

- Click on System Preferences >> Phidgets (under Other) to activate the Preference Pane
- Make sure that the **Phidget High Speed Encoder** is properly attached.
- Double Click on **Phidget High Speed Encoder** in the Phidget Preference Pane to bring up the Encoder-full example. This example will function in a similar way as the Windows version.

If you are using Linux

There are no sample programs written for Linux.

Go to www.phidgets.com >> Drivers

Download Linux Source

- Have a look at the readme file
- Build Phidget21

The most popular programming languages in Linux are C/C++ and Java.

Notes:

Many Linux systems are now built with unsupported third party drivers. It may be necessary to uninstall these drivers for our libraries to work properly.

Phidget21 for Linux is a user-space library. Applications typically have to be run as root, or udev/hotplug must be configured to give permissions when the Phidget is plugged in.

If you are using Windows Mobile/CE 5.0 or 6.0

Go to www.phidgets.com >> Drivers

Download x86, ARMV4I or MIPSII, depending on the platform you are using. Mini-itx and ICOP systems will be x86, and most mobile devices, including XScale based systems will run the ARMV4I.

The CE libraries are distributed in .CAB format. Windows Mobile/CE is able to directly install .CAB files.

The most popular languages are C/C++, .NET Compact Framework (VB.NET and C#). A desktop version of Visual Studio can usually be configured to target your Windows Mobile Platform, whether you are compiling to machine code or the .NET Compact Framework.

Programming a Phidget

Phidgets' philosophy is that you do not have to be an electrical engineer in order to do projects that use devices like sensors, motors, motor controllers, and interface boards. All you need to know is how to program. We have developed a complete set of Application Programming Interfaces (API) that are supported for Windows, Mac OS X, and Linux. When it comes to languages, we support VB6, VB.NET, C#.NET, C, C++, Flash 9, Flex, Java, LabVIEW, Python, Max/MSP, and Cocoa.

Architecture

We have designed our libraries to give you the maximum amount of freedom. We do not impose our own programming model on you.

To achieve this goal we have implemented the libraries as a series of layers with the C API at the core surrounded by other language wrappers.

Libraries

The lowest level library is the C API. The C API can be programmed against on Windows, CE, OS X and Linux. With the C API, C/C++, you can write cross-platform code. For systems with minimal resources (small computers), the C API may be the only choice.

The Java API is built into the C API Library. Java, by default is cross-platform - but your particular platform may not support it (CE).

The .NET API also relies on the C API. Our default .NET API is for .NET 2.0 Framework, but we also have .NET libraries for .NET 1.1 and .NET Compact Framework (CE).

The COM API relies on the C API. The COM API is programmed against when coding in VB6, VBScript, Excel (VBA), Delphi and Labview.

The ActionScript 3.0 Library relies on a communication link with a PhidgetWebService (see below). ActionScript 3.0 is used in Flex and Flash 9.

Programming Hints

- Every Phidget has a unique serial number - this allows you to sort out which device is which at runtime. Unlike USB devices which model themselves as a COM port, you don't have to worry about where in the USB bus you plug your Phidget in. If you have more than one Phidget, even of the same type, their serial numbers enable you to sort them out at runtime.
- Each Phidget you have plugged in is controlled from your application using an object/handle specific to that phidget. This link between the Phidget and the software object is created when you call the .OPEN group of commands. This association will stay, even if the Phidget is disconnected/reattached, until .CLOSE is called.
- For full performance, the Phidget APIs are designed to be used in an event driven architecture. Applications that require receiving all the data streaming from the device will have to use event handlers, instead of polling.

Networking Phidgets

The PhidgetWebService is an application written by Phidgets Inc. which acts as a network proxy on a computer. The PhidgetWebService will allow other computers on the network to communicate with the Phidgets connected to that computer. ALL of our APIs have the capability to communicate with Phidgets on another computer that has the PhidgetWebService running.

The PhidgetWebService also makes it possible to communicate with other applications that you wrote and that are connected to the PhidgetWebService, through the PhidgetDictionary object.

Documentation

Programming Manual

The Phidget Programming Manual documents the Phidgets software programming model in a language and device unspecific way, providing a general overview of the Phidgets API as a whole. You can find the manual at www.phidgets.com >> Programming.

Getting Started Guides

We have written Getting Started Guides for most of the languages that we support. If the manual exists for the language you want to use, this is the first manual you want to read. The Guides can be found at www.phidgets.com >> Programming, and are listed under the appropriate language.

API Guides

We maintain API references for COM (Windows), C (Windows/Mac OSX/Linux), Action Script, .Net and Java. These references document the API calls that are common to all Phidgets. These API References can be found under www.phidgets.com >> Programming and are listed under the appropriate language. To look at the API calls for a specific Phidget, check its Product Manual.

Code Samples

We have written sample programs to illustrate how the APIs are used.

Due to the large number of languages and devices we support, we cannot provide examples in every language for every Phidget. Some of the examples are very minimal, and other examples will have a full-featured GUI allowing all the functionality of the device to be explored. Most developers start by modifying existing examples until they have an understanding of the architecture.

Go to www.phidgets.com >> Programming to see if there are code samples written for your device. Find the language you want to use and click on the magnifying glass besides "Code Sample". You will get a list of all the devices for which we wrote code samples in that language.

API for the PhidgetEncoder HighSpeed

We document API Calls specific to this product in this section. Functions common to all Phidgets and functions not applicable to this device are not covered here. This section is deliberately generic. For calling conventions under a specific language, refer to the associated API manual. For exact values, refer to the device specifications.

Functions

int EncoderCount() [get] : Constant

Returns the number of encoders supported by this PhidgetEncoder. The 1057 supports one optical encoder.

int Position(int EncoderIndex) [get,set]

Returns/sets the position of an encoder. This is an absolute position as calcutated since the encoder was plugged in. Dividing position by the number of increments per revolution will give the number of rotations the encoder has travelled.

Position can be set, typically used when an encoder has reached an identifiable (through external means, such as a limit switch) home position. This call does not send information to the device, as an absolute position is maintained only in the library. After this call, position changes from the encoder will use the new value to calculate absolute position

Events

OnPositionChange(int EncoderIndex, int Time, int PositionChange) [event]

An event that is issued whenever a change in encoder position occurs. This event returns the length of time that the change took (in milliseconds), and the amount of change (positive/negative encoder increments)

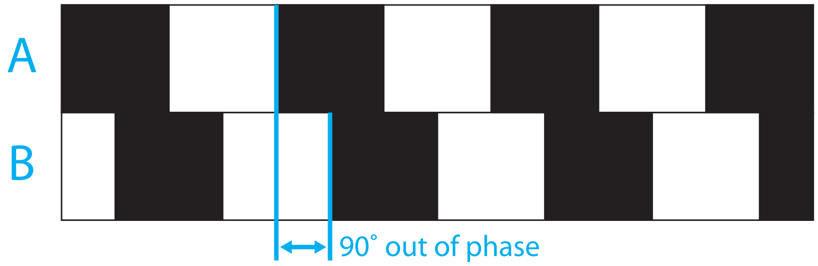
Technical Section

The PhidgetEncoder HighSpeed can be used with a wide assortment of mechanical and optical encoders. The encoder should be of quadrature output type, indicating that there will be two output channels (usually labeled A and B). Specifically, the PhidgetEncoder HighSpeed reads a standard incremental non-differential TTL encoder.

The maximum rate of the PhidgetEncoder is specified at 500,000 counts per second. In your application, this number relates directly to the number of revolutions per second you wish to measure, and the number of counts per revolution specified for your encoder. If your encoder measures 1000 counts per revolution, then the limit on measureable revolutions per second is 500.

Quadrature Encoder Fundamentals

Quadrature encoders are the most popular type of encoder, using two output channels to dictate both angular position and direction of rotation. The term quadrature relates to a system whose components exist at an angle of 90° to each other. In an encoder system, two parallel mechanical disks or optical gratings are set 90° out of phase. In this way, as the two disks spin in unison, the output can signify both the number of pulses that have occured (the angular position) as well as which output channel is leading the other (direction of rotation).



Choosing Encoders

Both mechanical and optical encoders are available, with optical encoders prevailing in quality at higher revolution speeds. Review the data sheet for the encoder that your are selecting carefully before purchasing it to ensure it is compatible with the PhidgetEncoder HighSpeed. Almost any incremental quadrature encoder will work but it is important to verify this before connecting it to the Phidget. Absolute encoders will not work with this device.

Warning: The PhidgetEncoder HighSpeed incorporates a 1kiloOhm pull-up resistor on each line from the encoder input connector. Some encoders will not be able to sink enough current to reliably signal to the 1057. They may work initially, or not at all.

We have reviewed the following encoders, and found that they can be used with the PhidgetEncoder HighSpeed. This is not meant to be a comprehensive list but should be used as a comparison with other encoders.

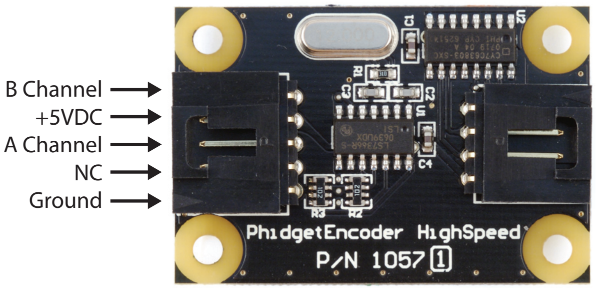
Manufacturer	**Web Page**	**Part Number**
Grayhill	www.Grayhill.com	Series 63R, Series 61R Series 63Q TTL Output Series 64T TTL Output
US Digital	www.USDigital.com	S4, S5S, E2, E3, E4, E4P, etc.
Avago Technologies (Formerly Agilent)	www.avagotech.com	HEDS 5500
CUI Inc.	www.cui.com	AMT103-V

Cable and Connector Components for Encoder Connector

Manufacturer	Part Number	Description
Molex	50-57-9405	5 Position Cable Connector
Molex	16-02-0102	Wire Crimp Insert for Cable Connector
Molex	70543-0004	5 Position Vertical PCB Connector
Molex	70553-0004	5 Position Right-Angle PCB Connector (Gold)
Molex	70553-0039	5 Position Right-Angle PCB Connector (Tin)
Molex	15-91-2055	5 Position Right-Angle PCB Connector - Surface Mount

Note: Most of the above components can be bought at www.digikey.com

Connector Pinout



Device Specifications

Characteristic	Value
Maximum Count Rate	500,000 counts / second
Internal Output Pull-Up Resistance	2.2kΩ
Min/Max USB Supply Voltage	4.75 - 5.25VDC
Encoder Input Low Voltage	<0.8V
Encoder Input High Voltage	>3.6V
Device Quiescent Current Consumption	20mA (no encoder attached)
Device Active Current Consumption	100mA max
Operating Temperature	0 - 70°C

Product History

Date	Board Revision	Device Version	Comment
January 2005		300	Product Release

Support

Call the support desk at 1.403.282.7335 9:00 AM to 5:00 PM Mountain Time (US & Canada) - GMT-07:00

or

E-mail us at: support@phidgets.com